WEST Search History

DATE: Thursday, June 12, 2003

Set Name side by side	Query	Hit Count	Set Name result set
	DB=USPT,PGPB,JPAB,EPAB,DWPI,TDBD; THES=ASSIGNEE; PLUR=YES; OP=ADJ		
L10	l9 same ((pending or granted) with lock)	0	L10
L9	reliability with (field or column) same lock$3	110	L9
L8	parameterized same (lock$3 or mode) same(grant$3 or pending)	5	L8
L7	parametrized lock	0	L7
L6	L5 not l2	15	L6
L5	lock$3 with manag$3 same (grant$3 or pending) same (field or column) same (parameter or value or attribute)	17	L5
L4	lock with manag$3 same (grant$3 or pending) same (field or column) same (parameter or value or attribute)	17	L4
L3	5983225[uref]	4	L3
L2	l1 same ((pending lock or granted lock) same (memory or database or storage) same (request$3 or access$3))	19	L2
L1	resource same lock$3 or manag$5 same (field or column)	32472	L1

END OF SEARCH HISTORY

WEST

Generate Collection | Print

L2: Entry 12 of 19 File: USPT Jul 6, 1999

US-PAT-NO: 5920872
DOCUMENT-IDENTIFIER: US 5920872 A

TITLE: Resource management using resource domains

DATE-ISSUED: July 6, 1999

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Grewell; Patricia	San Mateo	CA		
Hayes; Terry N.	Los Altos	CA		
Bridge; William	Alameda	CA		
Karten; Hans	Ermelo			NL

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Oracle Corporation	Redwood Shores	CA			02

APPL-NO: 08/ 672348 [PALM]
DATE FILED: June 25, 1996

INT-CL: [06] G06 F 17/30

US-CL-ISSUED: 707/202; 707/204, 707/205
US-CL-CURRENT: 707/202; 707/204, 707/205

FIELD-OF-SEARCH: 395/650, 395/674, 395/726, 707/206, 707/204, 707/202, 707/205

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
5161227	November 1992	Dias et al.	
5403639	April 1995	Belsan et al.	395/650

OTHER PUBLICATIONS

"A distributed lock Manager on faul tolerant MPP" Aldred, M., Gertner, I.; McKellars, S. (Encore Comput. Corp. Marlborough, MA, USA), 1995.

ART-UNIT: 275

PRIMARY-EXAMINER: Toplu; Lucien U.

ASSISTANT-EXAMINER: Nguyen; Hanh

ATTY-AGENT-FIRM: McDermott, Will & Emery

ABSTRACT:

A method and apparatus for managing access to resources is provided. When a process requires access to a resource, the process requests a lock on the resource from a lock manager unit that resides on the same node as the process. If a resource object for the resource does not exist, one is created in the lock manager unit, but not on lock manager units on other nodes. Because each lock manager unit does not have to store all resource objects, and resource objects are only created for resources that are actually used, the overhead of the lock management system is significantly reduced. Resources are grouped in recovery domains. When a lock manager unit that supported a recovery domain fails, the recovery domain is marked invalid. All resources in the recovery domain are considered invalid unless it would have been impossible for a failed instance to have held an exclusive lock on the resource. A snapshot of lock information is made before cleanup is performed on invalid resources. After cleanup, the snapshot is used to determine which resources were cleaned up.

12 Claims, 7 Drawing figures

WEST

Generate Collection | Print

L6: Entry 6 of 15 File: USPT Apr 28, 1998

US-PAT-NO: 5745747
DOCUMENT-IDENTIFIER: US 5745747 A

TITLE: Method and system of lock request management in a data processing system having multiple processes per transaction

DATE-ISSUED: April 28, 1998

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Chang; Amy	Cortlandt Manor	NY		
Hsiao; Hui-I	Yorktown Heights	NY		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
International Business Machines Corporation	Armonk	NY			02

APPL-NO: 08/ 769845 [PALM]
DATE FILED: December 20, 1996

PARENT-CASE:
This application is a continuation of application Ser. No. 08/384,503, filed on Feb. 6, 1995, which was abandoned upon the filing hereof.

INT-CL: [06] G06 F 17/30

US-CL-ISSUED: 395/608; 395/726
US-CL-CURRENT: 707/8; 710/200

FIELD-OF-SEARCH: 395/600, 395/650, 395/608, 395/672, 395/674, 395/676, 395/726, 364/DIG.1, 364/DIG.2

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4716528	December 1987	Crus et al.	395/650
☐	4897782	January 1990	Bennett et al.	395/600
☐	5062038	October 1991	Jordan, Jr.	395/650
☐	5063501	November 1991	Jordan, Jr.	395/725
☐	5063504	November 1991	Jordan, Jr.	395/725
☐	5117352	May 1992	Falek	395/575
☐	5161227	November 1992	Dias et al.	395/650
☐	5247672	September 1993	Mohan	395/650
☐	5355477	October 1994	Strickland et al.	395/600
☐	5414839	May 1995	Joshi	395/600
☐	5485607	January 1996	Lomet et al.	395/600

ART-UNIT: 237

PRIMARY-EXAMINER: Black; Thomas G.

ASSISTANT-EXAMINER: Von Buhr; M. N.

ATTY-AGENT-FIRM: Galasso; Raymond M. Jenkins & Gilchrist Ludwin; Richard M.

ABSTRACT:

A data processing system and method which manages lock requests at both the transaction and process levels. A lock manager allocates one lock request block (LRB) per process and identifies the ownership of the lock by storing process, transaction and shared IDs in the LRB. The process ID is used to identify which lock to release when a process wants to release the lock before transaction commit while the transaction ID is used to identify all locks to be released at the commit/roll back time of the transaction. The shared ID is a unique value which identifies the lock. The LRB also has a granted transaction mode (GTM) field which stores a value representing the upper bound of the lock modes granted to the transaction. Whenever a lock is granted or released, the GTM of the transaction is recomputed.

16 Claims, 3 Drawing figures

WEST

Generate Collection | Print

L2: Entry 13 of 19 File: USPT Aug 23, 1994

US-PAT-NO: 5341491
DOCUMENT-IDENTIFIER: US 5341491 A

TITLE: Apparatus and method for ensuring that lock requests are serviced in a multiprocessor system

DATE-ISSUED: August 23, 1994

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Ramanujan; Raj	Leominster	MA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Digital Equipment Corporation	Maynard	MA			02

APPL-NO: 08/ 141427 [PALM]
DATE FILED: October 22, 1993

PARENT-CASE:
This application is a continuation of application Ser. No. 07/546,365, filed Jun. 29, 1990.

INT-CL: [05] G06F 12/14

US-CL-ISSUED: 395/425; 366/DIG.1, 366/244.3, 366/246.8, 366/228.1, 366/230.2, 366/280.8, 366/284.1, 366/284.5
US-CL-CURRENT: 711/152; 711/147

FIELD-OF-SEARCH: 395/400, 395/425, 364/2MSFile, 364/9MSFile

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
4897786	January 1990	Pimm et al.	395/325
4937733	June 1990	Gillett, Jr. et al.	395/325
5050066	September 1991	Myers et al.	395/575
5060144	October 1991	Sipple et al.	395/650
5068781	November 1991	Gillett, Jr. et al.	395/325
5161227	November 1992	Dias et al.	395/650
5175837	December 1992	Arnold et al.	395/425

OTHER PUBLICATIONS

Kelter, Udo, Information Systems "Deadlock-Freedom of Large Transactions In Object Management Systems," vol. 14, No. 2, pp. 175-180, (1989) Dortmund, Germany.
Anderson et al., Performance Evaluation Review, "The Performance Implications of Thread Management Alternatives For Stored Memory Multiprocessors," vol. 17, No. 1, pp. 49-60, (May 1989) Berkeley, USA.

ART-UNIT: 232

PRIMARY-EXAMINER: Dixon; Joseph L.

ASSISTANT-EXAMINER: Nguyen; Hiep T.

ATTY-AGENT-FIRM: Paciulan; Richard J. Maloney; Denis C.

ABSTRACT:

A lockout avoidance circuit is provided for a plurality of nodes which generate lock requests for a shared resource such as a memory location. The circuit insures that lock requests are eventually satisfied. A lock queue includes a plurality of registers pipelined together. Lock requests only enter the lock queue if they are refused access to a shared resource a predetermined number of times. A first register is the head of the queue and the last register is the bottom of the queue. An enabling circuit allows the queue to store in the registers lock requests received from the different nodes in the order in which they are initially refused service. The enabling circuit operates the queue by pushing the stored lock requests toward the head of the queue each time the head entry in the queue is serviced. The lockout avoidance circuit is implemented at each level of the system wherein a lockout condition can occur.

22 Claims, 7 Drawing figures

L8: Entry 5 of 5 File: TDBD Mar 1, 1977

TDB-ACC-NO: NN77033887

DISCLOSURE TITLE: Locking Protocols for Concurrent Operations on B Trees. March 1977.

PUBLICATION-DATA:
IBM Technical Disclosure Bulletin, March 1977, US

VOLUME NUMBER: 19
ISSUE NUMBER: 10
PAGE NUMBER: 3887 - 3889

PUBLICATION-DATE: March 1, 1977 (19770301)

CROSS REFERENCE: 0018-8689-19-10-3887

DISCLOSURE TEXT:

3p. A mechanism which allows concurrent use of a B-tree structure is presented. A protocol for using this mechanism is also included. The protocol can be shown to be deadlock-free and is parameterized so that maximum concurrency can be achieved for a given mixture of users of the structure. - A B-tree 1ù is a dynamic data structure which allows users to maintain and retrieve elements from a sorted set when deletions and insertions are frequently made to the set. The structures have wide application as support of access paths to indexed files 2ù. For simplicity, we will use the variant presented by Wedekind 3ù. However, the ideas presented here are easily extended to any other type of B-tree. When these structures are used in a multiuser environment, they may cause a bottleneck in the operation of the system in which they exist since all users have to access them, and without a proper synchronization mechanism available, all the accesses to it have to be serialized. Although B-trees are well suited for access by a single user, no satisfactory mechanism for concurrent access has been previously reported, the best one being that presented in 4ù. - The mechanism proposed here has four different signals (RR, RU, A, X) that users may give to each other. These signals (which we will call locks) are to be placed in the nodes of the structure being visited. Depending on the type of each lock placed on a node by a user, access to the node in question may be prohibited or restricted for other users. This control over access to a node is enforced by restricting the types of locks that can be simultaneously placed in any given node. These restrictions, called incompatibilities, are shown below. - Existing lock RR RU A X Lock RR yes yes yes no to RU yes yes no no be A yes no no no placed x no no no no Compatibilities Among Locks This table is to be interpreted as follows: If a user wishes to visit a node of the structure, he has to place a lock on it. If the lock to be placed is compatible (as shown by a yes entry in the table) with every existing lock on the node, access is permitted and the new lock is placed in the node. If the locks are incompatible, however, the incoming user is placed on a wait set for visiting this node. Once a user is placed on a wait set for a node, he cannot request that locks be placed on other nodes. (The way users are selected for removal from these sets is not important here. Any reasonable method can be used, e.g., a round-robin selection). - With the above synchronization mechanism, we now describe protocols to be used by users of these data structures. Two types of users are distinguished: those that will access the structure to retrieve information only and those that wish to modify it. We call the first type of users readers and the second type updaters. Modification to these structures (either by

way of an insertion or a deletion) is a known process and will not be described here (see 3ù). - Protocol for a reader: We note that access to a B-tree structure by a reader starts at the root of the structure and proceeds toward a leaf. In order to visit a node, a reader attempts to place an RR lock. Once the access is granted, the reader releases any other lock it may have and proceeds to visit the node. - Protocol for an updater: Access to these structures by updaters is done in two phases. First, a search occurs starting at the root and ending at a leaf node. Then the actual alteration of the structure is done, starting at a leaf node and proceeding toward the root (see 3ù for details). - Before an updater accesses the structure, two parameters are selected: p and q. Let the height of the tree be h. Define q = min (h,q), p = min (p, h- q), r = h- p - q. - In the search phase, an updater places up to p RU locks, starting from the root down proceeding, as a reader does, to release previously held locks when granted access to a node. For the next r levels down, the lock requested before an access to a node is made is an A lock. If the access is granted, the node is visited. While visiting the node, the user can determine if the parent of this node will be affected by the update (see 5ù for details). If the parent node will not be affected by the update, all previously held locks are released. Finally, for the last q levels, X locks are used, again releasing locks as when placing A locks. - If, upon reaching a leaf node, the user still has an RU lock placed on some node, he releases all his nodes and has to make another attempt to update the structure, starting at the top. If the previous condition does not occur but the user still has A locks, he then first converts all his X locks into A locks and then converts, starting with that node closest to the root on which he has an A lock, all the A locks into X locks. If, when attempting this conversion, an X lock cannot be granted, the user retains its A lock on that node and waits until the conversion can proceed. - Once this last conversion step is done, or if the updater did not have any A locks to convert, the updater will be holding only X locks only in those nodes that will have to be modified by the update. The modification can now be done and the X locks released when completed. - It is shown in 5ù that this technique gives, for proper choice of the parameters p and q, a maximum concurrency of accesses to the structure with a minimum of overhead. The protocols also guarantee that no deadlock situation can occur during access to the B-tree, thus eliminating the need for dynamic backout of users involved in such a situation. References 1ù Bayer, R. and McCreight, E., "Organization and Maintenance of Large Ordered Indexes," Acta Informatica 1, 3 (1972), 173-189. 2ù Astrahan, M. M., et al, "System R:Relational Approach to Database Management," ACM Transactions on Data Base Systems, Vol. 1, No. 2, June 1976, pp. 97-137. 3ù Wedekind, H., "On the Selection of Access Paths in a Data Base System," (Klimbie, J. W., Koffeman, K. L., (eds.)) Data Base Management, North-Holland, Amsterdam, 1974, pp. 385-397. 4ù Metzger, J. K., "Managing Simultaneous Operations in Large Ordered Indexes," TUM-Math. Report, Technical University of Munich, Institut fur Informatik, 1975. 5ù Bayer, R. and Schkolnick, M., "Concurrency of Operations on B-Trees," IBM Res. Report RJ 1791, May 1976.